

# Detecting Novel Processes with CANDIES – An Holistic Novelty Detection Technique based on Probabilistic Models

Christian Gruhl, Bernhard Sick

arXiv:1605.05628v1 [cs.LG] 18 May 2016

**Abstract**— In this article, we propose CANDIES (Combined Approach for Novelty Detection in Intelligent Embedded Systems), a new approach to novelty detection in technical systems. We assume that in a technical system several processes interact. If we observe these processes with sensors, we are able to model the observations (samples) with a probabilistic model, where, in an ideal case, the components of the parametric mixture density model we use, correspond to the processes in the real world. Eventually, at run-time, novel processes emerge in the technical systems such as in the case of an unpredictable failure. As a consequence, new kinds of samples are observed that require an adaptation of the model. CANDIES relies on mixtures of Gaussians which can be used for classification purposes, too. New processes may emerge in regions of the models' input spaces where few samples were observed before (low-density regions) or in regions where already many samples were available (high-density regions). The latter case is more difficult, but most existing solutions focus on the former. Novelty detection in low- and high-density regions requires different detection strategies. With CANDIES, we introduce a new technique to detect novel processes in high-density regions by means of a fast online goodness-of-fit test. For detection in low-density regions we combine this approach with a 2SND (Two-Stage-Novelty-Detector) which we presented in preliminary work. The properties of CANDIES are evaluated using artificial data and benchmark data from the field of intrusion detection in computer networks, where the task is to detect new kinds of attacks.

**Index Terms**—ovelty Detection Gaussian Mixture Models CANDIES Online Goodness-of-Fitovelty Detection Gaussian Mixture Models CANDIES Online Goodness-of-FitN

## 1 INTRODUCTION

Today, so-called “smart” or “intelligent” technical systems are often equipped with abilities to act in real environments that are termed to be “dynamic” in the sense that their characteristics are time-variant (change over time). But typically, knowledge about the basic nature of these changes is built into these systems and it is assumed that only the time when these changes occur cannot be predicted. Future systems, however, have to evolve over time. Not all knowledge about any situations the system will face at run-time will be available at design-time. That is, the system has to detect and react on fundamental changes in *time-variant environments*. As

an example, we may consider technical systems that make observations of their environment with sensors and classify these observations (samples). In a *time-invariant environment*, there may be different causes for different kinds of observations (called “processes” in the following). The data are modeled (e.g., by means of probabilistic models such as Gaussian mixtures) and then a classifier is built (e.g., by gradually assigning components of the Gaussian mixture to classes). At run-time, the data model and the classifier are not adapted.

An example for such a system could be a machine, that produces various parts (cf. left part of Figure 1). A *process* in this hypothetical environment would be similar to the production of a certain part. If the system is monitored with multiple sensors (e.g.,  $S1$  and  $S2$ ), the resulting sensor signals span a two-dimensional *input space* (as shown in the middle part of Figure 1). Each value pair is called an *observation* or *sample*. With suitable machine learning techniques we can approximate the resulting distribution of *samples*. On the right side of Figure 1 a Gaussian mixture model (GMM, cf. 4.1) is used to approximate, and thus modeling, the sample distribution. Ideally, each component of the GMM describes a physical *process* in the environment. Reasons to rely on GMM for this purpose is that arbitrary continuous densities can be approximated by GMM (with any desired precision, based on the number of components) and the generalized central limit theorem, which states that the sum of i.i.d. random samples tends to be normally distributed (assumed, that the variance is finite). In technical system this is frequently the case, since observed sensor values are often the outcome of various random parameters that influence each other. In a *time-variant environment*, there may eventually be some conspicuous samples (cf. Fig. 2). Then (if the system is able to detect such a situation), some questions come up: Are these samples outliers of existing processes or not? If not, is there an anomaly in the observed environment or did a *new* process emerge that was unknown at design-time? And how can we build *novelty detection* methods to identify such *novel* processes? How can we adapt the data model to suit the changed environment and when (in order to find

C. Gruhl and B. Sick are with the University of Kassel, Department of Electrical Engineering and Computer Science, Wilhelmshoeher Allee 73, 34121 Kassel, Germany (email: cgruhl,bsick@uni-kassel.de).

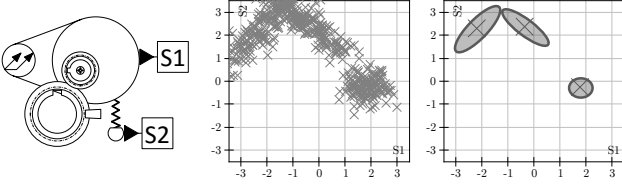


Fig. 1. Hypothetical scenario of a monitored machine. On the left: abstract machine with monitoring sensors S1 and S2. In the middle: the two-dimensional *input space* consisting of measured sensor signals from S1 and S2. In this case, the outcomes of three different processes are gathered in three clusters. On the right: approximated density model, the ellipses are called *components* and correspond to multivariate Gaussians that represent the physical *processes*.

a trade-off between fast and accurate reaction)? And if there are new model components, to which class do we have to assign them?

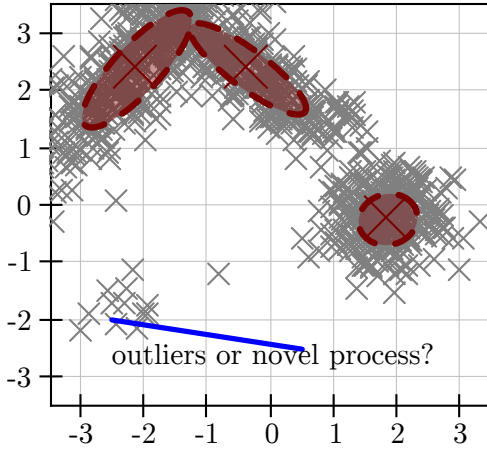


Fig. 2. A situation with samples produced by three processes (represented by three components and marked in red) and a model of the situation with three components. Some observations appear in a *low-density region* and are “suspicious”, they may indicate that a novel process currently emerges.

With CANDIES (Combined Approach for Novelty Detection in Intelligent Embedded Systems), some of the questions can be answered. One major challenge is the reliable detection of (possibly multiple) *novel processes* in the complete input space. Herby we assume that the input space is divided into two parts:

- 1) *High-density regions* (HDR): These are regions that are already covered by one or more components of the mixture model (i.e., the support of the kernels, in our case Gaussians, is high). This implies that *normal* observations are expected to appear in these regions and thus forming the *normal model*. However, new processes might also emerge here (e.g., “close” to, or between existing components, or even

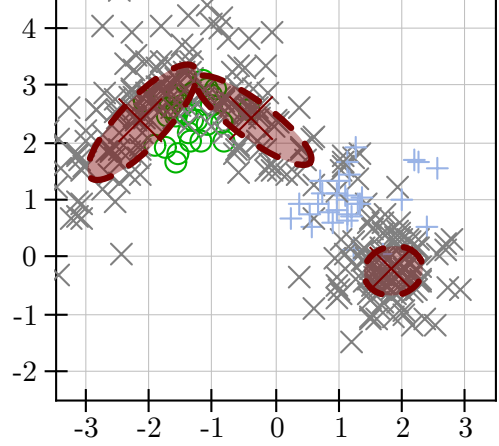


Fig. 3. Same situation with samples produced by three processes (represented by three components and marked in red) and corresponding model. Some observations (green circles  $\circ$  and blue crosses  $+$ ) in the *high-density region* covered by the model components, are the outcome of two not yet known processes and should be considered to be “suspicious”.

totally overlapping these) and therefore change the characteristics of the approximated density. We assume that HDR can be considered to be spatially compact in the input space, and that they contain the majority of the overall density mass.

- 2) *Low-density regions* (LDR): These regions are distant from any component centers, resulting in a low support of the kernels. Thus, *normal* data is not expected to be observed here and observations appearing here are considered to be *suspicious*. In contrast to HDR we assume, that LDR are widely spread in the input space and that usually only a single LDR exists (i.e., not separated by HDR).

The transition between HDR and LDR is not strictly defined and is application dependent. Caused by their different characteristics, different problems are faced to detect *novel processes*. Since LDR have a potentially infinite support, the main difficulty is to efficiently find spatial relations (i.e., clusters) between *suspicious observations*. On the other hand, for HDR, two issues must be addressed: 1) which observations are assumed to be *normal* (outcome of an already known and modeled process) and which are *suspicious* (i.e., outcome of a *novel process*, or anomalies). 2) When is a *novel process* present.

In a preliminary article (see [15]), we presented 2-SNDR, an approach to solve the *novelty detection* problem sketched above for situations, where novel processes start to “generate” data in LDR of a probabilistic knowledge model (based on Gaussian mixtures). Figure 2 depicts such an exemplary scenario (where a *novel process* is emerging in a LDR). To detect *novelty* in HDR CANDIES relies on a new approach that is premised on statistical goodness-of-fit testing (i.e., measuring how

well observed samples fit the assumed distribution), adjusted to suite Gaussian mixture models (GMM, cf. Section 4.1) and online environments. Figure 3 shows a different situation, where two *novel processes* started to “generate” samples in a HDR, but are not yet represented in the current model.

Altogether, it is possible to address a specific kind of time-variance in the observed environment which is useful for many applications. We may imagine other kinds of situations where processes disappear (*obsoleteness*) or change some basic parameters (*concept shift* or *concept drift*). Our current research addresses these situations as well.

The remainder of this article is organized as follows: Section 3 gives a broad overview of related work, including other common novelty detection techniques, related topics, and what are the distinctions to CANDIES. Section 4 briefly summarizes methodical foundations essential for this article. Preliminary to the technical in depth details, a simplified overview of the idea behind the proposed technique is given in Section 2. The main body, introducing CANDIES in detail, is contained in Section 5. In Section 6 a small case study based on the KDD Cup 99 Computer Intrusion data set is presented. Finally, a conclusion and outlook to future work is given in Section 7.

## 2 OVERVIEW OF CANDIES

With CANDIES we aim on three main goals: 1) Detecting clusters of *suspicious* samples (i.e., those that differ notably from what is expected). 2) Detecting such clusters in the complete input space, that is, in LDR and in HDR. 3) using the discovered clusters to model new *processes*.

The algorithms consists of multiple detectors for HDR and a single one for the LDR. It works (simplified) in the following manner. The foundation of the whole approach is a GMM, that provides a density estimate of the expected data. An advantage of GMM is that they can easily be extended to a classifier and that they belong to the family of generative models, thus additional structural information about the expected observations can be deduced (in contrast to discriminative classifiers, e.g., SVM). At first a new sample  $x'$  located either in a HDR or in a LDR. Depending on the location it is marked as *normal* (located in HDR) or *suspicious* (located in LDR) (*suspicious* is what comparable algorithms mark as *novelties*). Depending on that decision either the LDR detector or one of the HDR detectors is responsible for handling the new sample. If the sample is marked as *suspicious* the sample is stored in a ring buffer on which a nonparametric clustering is performed. If a cluster in the buffer reaches a certain size the detector will report the detection of a *novel process*. Otherwise, when the sample is regarded as *normal*, it is used to update one of the HDR-Detectors (there is one HDR-Detector for each individual component of the GMM), the decision

which detector is updated is made at random. The HDR-Detector works by testing how well the last  $m$  samples are fitting the estimated Gaussian bell. This is done by using a  $\chi^2$  test. If the  $t$ -value exceeds the critical value the detector reports the detection of a *novel process*.

## 3 RELATED WORK

The main task for a Novelty Detector is to distinguish if a previously unseen *sample* belongs either to a *normal* model or if it is *different* in some way so that it does not belong to the *normal* data and is therefore *novel*. Closely related to the topic are the fields of anomaly and outlier detection. Over a decade ago it was sufficient to roughly group novelty detection approaches into two classes: either statistical (cf. [24]) or neural network based (cf. [25]).

Most of the statistical approaches are relying on a model of the processed data. Observations are identified as (potentially) *novel* if they differ too much from what is expected, e.g., described by an appropriate model. Further, these approaches can be discerned based on the models they are using – either parametric or nonparametric models. Novelty detection techniques based on nonparametric density modeling are, for example, those using  $k$ -nearest neighbors approaches or kernel density estimators, see [37] for a sample application in intrusion detection. Parametric models on the other side make assumptions about the distribution of the observed samples, e.g. Gaussian mixture models. In preliminary work [12] we detect *novelty* based on a parametric Gaussian mixture model and a state variable which monitors how well the observations fit the model. The approach is used for comparison to CANDIES and briefly presented in the case study in Section 6.

The second group comprises detection techniques that are based on neural networks, e.g., multi-layer perceptrons, radial basis function neural networks, [4], [6] but, according to Markou and Singh [25], also include methods based on support vector machines, e.g. One-Class SVM as described by Tax and Duin [35].

Since the early 2000s, the topic draw much attention as objective of research and changed considerably (i.e. new ranges of applications or whole new techniques, due to advances in computing power). Now, a more recent survey [29] suggest five different categories to group novelty detection approaches: **i)** probabilistic, **ii)** distance-based, **iii)** reconstruction-based, **iv)** domain-based, and **v)** information theoretical.

The first category covers a large part of the approaches that were previously affiliated with the *statistical* group. Typically these techniques are build upon a parametric density estimation of training data as a model. Frequently used are mixtures of Gaussians ([12] [20], [38], for instance). Novelty is usually detected if samples are observed in *low-density-regions* (i.e., the density for the observed sample is below a selected threshold). Several method to define a threshold are based on Extreme

Value Theory (EVT, cf. [8], [18], [32]). The idea in EVT is to estimate the distribution of extreme values (i.e., maximum or minimum for legit samples) for a given density model and a given sample size. Then, samples that exceed the expected maximum or surpass the expected minimum are identified as *novel*. Recently Extreme Learning Machines with decision making depending on EVT where proposed by Al-Behadili *et al.* [2] to implement incremental semi-supervised learning based on novelty detection. Thus, probabilistic approaches are not limited to generative models, cf. [10], for example, where Support Vector Machines are used for detection and resulting novelty values calibrated in order to be interpreted as class-conditional probabilities.

To the second category belong approaches that are based on distances. Popular representatives of this category are approaches based on  $k$ -nearest-neighbors ( $knn$ ). E.g., [7] or [17], [27], where the latter use the density of a  $k$ -neighborhood (i.e., a radius required to enclose  $k$  neighbors) to identify *novel* samples. A sample is *novel* if its neighborhood density is considerably lower than the density of its neighbors. Clustering based approaches refer also to category ii). Typically, *normal* samples are aggregated to form clusters, *novelty* is then determined by the minimal distance of an unseen sample to any centroid (e.g. [33], [36]). It is questionable whether category i) and ii) are sharply differentiable. Gaussian Mixture Models for example, consists of multiple location invariant kernels and the density is finally greatly dependent on the applied distance measure.

Our new CANDIES approach does not fit into a single category but is a hybrid in the sense that it belongs to the first two categories: probabilistic and distance-based. For a detailed summary of the remaining categories iii), iv), and v) cf. [29].

However, most of the introduced paradigms are designed to spot only single samples as *novelties* and do not relate those samples to one another. Thus, potential new knowledge (structural information in form of a cluster, that is evidence of a *novel physical process*) is unexploited and discarded. In some common applications such as medical condition monitoring [9], [31], [34] or machinery monitoring [30], this is not a real drawback, since *anomalies* might arise everywhere in the input space and are very specifically stuck to a concrete application (i.e., monitoring a specific patient or a specific engine). But in other fields, such as network intrusion detection, this discovered *knowledge* has great potential to be used to detect future attacks.

The contributions of this article are:

- 1) CANDIES is trimmed to detect *novel* processes (clusters of *suspicious* observations, cf. *knowledge*) in such a way, that the process can easily be integrated as new component into the existing GMM. This leads to the result, that *learning* does not only happen in a isolated training phase, off-line at design-time, but it is also conducted at run-time [16].

- 2) Novelty is not only detected in *low-density* regions (where *normal* observations are not likely to appear), but also in *high-density* regions, i.e., where *normal* observations are expected.

## 4 METHODOLOGICAL FOUNDATIONS

For the purpose of a self-contained article, we briefly recap the most important techniques used to implement our approach. This includes an overview of Gaussian mixtures, a method to extend those to a classifier, a short introduction to nonparametric density estimation as foundation for cluster analysis, and a brief description of statistical goodness-of-fit testing.

### 4.1 Gaussian Mixtures

One frequently used approach to generative modeling is the Gaussian mixture model (GMM). That is, a superposition of multiple multivariate normal distributions (denoted as  $\mathcal{N}$  and commonly referred to as Gaussian) and a mixing coefficients  $\pi_j$  (Eq. (1)). Each Gaussian is called a *component* and has its own set of parameters which are the mean vector  $\mu_j \in \mathbb{R}^D$  and a covariance matrix  $\Sigma_j$  (with  $\dim(\Sigma_j) = D \times D$ ) that describes its shape. The mixing coefficients  $\pi_j$  (with constraints  $\sum_{j=1}^J \pi_j = 1$ ,  $\pi_j \in \mathbb{R}^+$ ) ensure that the resulting  $p(\mathbf{x})$  ( $\mathbf{x} \in \mathbb{R}^D$  is the random variable) still fulfills the requirements for a density function. They may also be interpreted as priors for each component (i.e, the probability that an unobserved sample is generated by the corresponding component). Altogether, we get:

$$p(\mathbf{x}) = \sum_{j=1}^J \pi_j \cdot \mathcal{N}(\mathbf{x}|\mu_j, \Sigma_j). \quad (1)$$

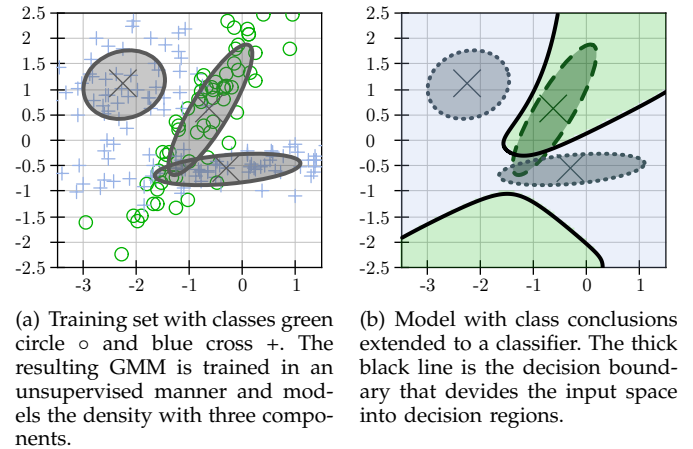


Fig. 4. Each  $\times$  denotes the center  $\mu_j$  of the  $j$ -th component while each ellipse represents the shape which is defined by the  $j$ -th covariance matrix  $\Sigma_j$ . The distance between  $\mu_j$  to the associated ellipse, which is a constant density surface, corresponds to a Mahalanobis distance of 1.

An ordinary GMM models only the density of an associated training set and can be trained in an unsupervised manner (i.e., labels are not required). Since the sufficient statistics for the components cannot be computed in closed form, we pursue this goal with an expectation-maximization (EM) like approach that uses 2nd order (or hyper-) distributions and is heavily based on variational Bayesian inference (VI). An extensive introduction to VI is given by [5]. For clarification, a trained GMM for a two-dimensional data set is shown in Figure 4(a).

Relying on VI gives rise to two advantages: (1) prior knowledge about the data can be included, which is especially valuable in real-world applications, and (2) multiple GMM can be *fused* into one model as described in [14]. The final GMM is obtained from the expectations (a point estimate from the second-order distributions) of the hyper-distributions after the VI training finishes.

Since we assume a certain functional form of the underlying distribution and estimate its parameters, GMM are parametric density models.

## 4.2 Classification Paradigm

To derive a classifier  $h(\mathbf{x})$  from the trained density model  $p(\mathbf{x})$ , we estimate the class posteriors  $p(c|\mathbf{x})$  in a second, supervised (i.e., with respect to class labels) iteration. The classification of a given sample  $\mathbf{x}$  is then done, as shown in Eq. (2) by selecting the maximum a-posteriori (MAP) of the class probabilities:

$$h(\mathbf{x}) = \underset{c}{\operatorname{argmax}} \{p(c|\mathbf{x})\}, \quad (2)$$

with

$$p(c|\mathbf{x}) = \sum_{j=1}^J p(c|j) \cdot p(j|\mathbf{x}) = \sum_{j=1}^J \xi_{j,c} \cdot \gamma_{\mathbf{x},j}, \quad (3)$$

where

$$\gamma_{\mathbf{x},j} = \frac{\pi_j \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{j'=1}^J \pi_{j'} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_{j'}, \boldsymbol{\Sigma}_{j'})}, \quad (4)$$

$$\xi_{j,c} = \frac{1}{N_j} \sum_{\mathbf{x}_n \in \mathbf{X}_c} \gamma_{\mathbf{x}_n,j}. \quad (5)$$

Eq. (4) shows the *responsibilities*  $\gamma_{\mathbf{x},j}$  which are the probability that a given sample  $\mathbf{x}$  was generated by the  $j$ -th component. For each component  $j$  and class  $c$  the conclusion is determined by Eq. (5), which is the fraction of all responsibilities for samples  $\mathbf{x}_n \in \mathbf{X}_c$  that are labeled with class  $c$  and the effective number of samples (denoted as  $N_j = \sum_{n=1}^N \gamma_{\mathbf{x}_n,j}$ ) belonging to the  $j$ -th component ( $\mathbf{X}$  is the overall set of labeled samples,  $\mathbf{X}_c$  the subset of  $\mathbf{X}$  associated with class  $c$ ).

Finally, the class posteriors  $p(c|\mathbf{x})$  given in Eq. (3) are a composition of the responsibilities  $\gamma_{\mathbf{x},j}$  and the class conclusions  $\xi_{j,c}$ . The resulting decision boundary, which describes the classifier for the previously estimated density model, is shown in Figure 4(b).

## 4.3 Density Based Clustering

Rather than assuming a specific functional form such as parametric methods, nonparametric techniques provide a point estimate for the density  $p(\mathbf{x})$  at a given point  $\mathbf{x}$ . One well-known nonparametric method is the Parzen window (or kernel) density estimator, here with Gaussian kernel:

$$p(\mathbf{x}) = \frac{1}{N \cdot h^D} \sum_{n=1}^N k\left(\frac{\mathbf{x} - \mathbf{x}_n}{h}\right). \quad (6)$$

It is the sum of a finite set of  $N$  samples  $\mathbf{x}_n$  of an underlying training set to which an appropriate *kernel function* is applied to. The kernel is placed at the point  $\mathbf{x}$  where the density should be estimated. The parameter  $h$  is a *smoothing* factor that controls how *smooth* the estimation is while  $D$  is the number of dimensions. Closely related to the Parzen window are histograms (cf. [5]).

The DBSCAN clustering algorithm (cf. [11]) uses a density estimation that is quite similar to a Parzen window estimator. Based on the density at each sample the algorithm decides whether a sample belongs to, lies at the edge, or is outside a cluster (in that case it is considered as *noise*). To do so, the kernel in Eq. (7):

$$k(\mathbf{x}) = \begin{cases} 1, & \text{if } \text{dist}(\mathbf{x}, \mathbf{0}) \leq \epsilon \\ 0, & \text{otherwise} \end{cases} \quad (7)$$

is used which forms an  $D$ -dimensional sphere around the point  $\mathbf{x}$  with radius  $\epsilon$ . Typically, *dist* is realized with an Euclidean metric. If a sample is part of a cluster, all samples inside the sphere are also assigned to the same cluster. The advantage of this approach is that clusters of arbitrary shapes can be identified.

## 4.4 Statistical Goodness-of-Fit Tests

To validate whether an observed sample matches a hypothesized distribution or not, goodness-of-fit tests can be applied. A fast and reliable method is Pearson's chi-squared ( $\chi^2$ ) test [28]. The test compares observed frequencies from mutually exclusive events (finite set of possible outcomes/values of a discrete random variable) against expected theoretical frequencies (obtained from a suitable fitted distribution) of these events. The test statistic (or  $t$ -value) is calculated by:

$$t = \sum_i^k \frac{(x_i - e_i)^2}{e_i} \quad (8)$$

where  $x_i$  is the observed event frequency of event  $i$  and  $k$  is the total number of different events. The expected frequencies of events  $i$  are given by  $e_i$ :

$$e_i = P_{\text{fit}}(i|\Theta). \quad (9)$$

where  $P_{\text{fit}}$  is the fitted distribution. The test aggregates the squared deviations between observed and expected frequencies and weights them by the expected frequency.



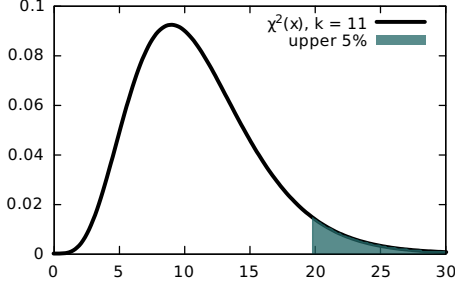


Fig. 5. Distribution of  $t$ -values for  $\chi^2$  test with 11 degrees of freedom. The marked region at the right is the rejection area for a significance level of  $\alpha = 5\%$ . The beginning of the region is equal to the critical value  $\chi_{5\%,11}^{2,upper}$ .

This leads to stronger penalties when only small frequencies are expected. To accept or reject the *null*-hypothesis that the sample is drawn from the hypothesized distribution the  $t$ -value must be less than the critical value:

$$\chi_{\alpha,k}^{2,upper} = F_{\chi^2}^{-1}(1 - \alpha) \quad (10)$$

The critical value is calculated by evaluating the inverse cumulative density function  $F^{-1}$  of the  $\chi^2$  distribution with  $\nu$  degrees of freedom at point  $1 - \alpha$ . Where  $\alpha$  is the *significance level* which implies that the error rate for type I errors is at most  $\alpha$  (often  $\alpha = 5\%$  or  $1\%$ ). The degrees of freedom  $\nu$  are given by the number of events minus the number  $p$  of covariate parameters  $\Theta$  of the fitted density  $P_{\text{fit}}$  (e.g.,  $p = 2$  for univariate Gaussian with  $\Theta = (\mu, \sigma)$ ). Figure 5 emphasizes the relation between the  $\chi^2$  distribution of  $t$ -values and the critical value  $\chi_{5\%,11}^{2,upper}$  for a significance level of  $\alpha = 5\%$ .

## 5 ONLINE NOVELTY DETECTION

This section is split into three parts: the first part is based on our previous work [15] and discusses novelty detection and reaction in LDR with 2SND. The second part treats novelty detection in HDR with online capable  $\chi^2$  goodness-of-fit tests. The last part then introduces CANDIES a detector which is able to detect novelties in the whole input space by combining both previously mentioned techniques. All techniques share the property to be applicable to online environments (i.e., soft real-time).

### 5.1 Novelty Detection in Low-Density Regions

To detect *novel processes* in sparse LDR we developed the 2 Stage Novelty Detection (2SND) algorithm. The algorithm works on top of an existing GMM or CMM (as described in Section 4.2) and extends it with novelty detection capabilities. Further, with 2SND it is possible to update the underlying GMM/CMM and to enhance them by including components that model the detected *novel processes*.

The algorithm itself consists of two procedures: a main procedure 2SND (Alg. 1) and an auxiliary procedure

PROPAGATE, that propagates the cluster id to all affiliated samples using a modified breadth-first search.

To detect *novel processes*, we propose a two-stage approach which identifies *suspicious* samples in the first stage and *novel processes* in the latter. Each assessed sample is individually tested how well it suits the current model by determining whether it resides in a high- (HDR) or low-density region (LDR). This is done by exploiting the fact that the squared Mahalanobis distances between samples from a Gaussian  $j$  to its mean  $\mu_j$ :

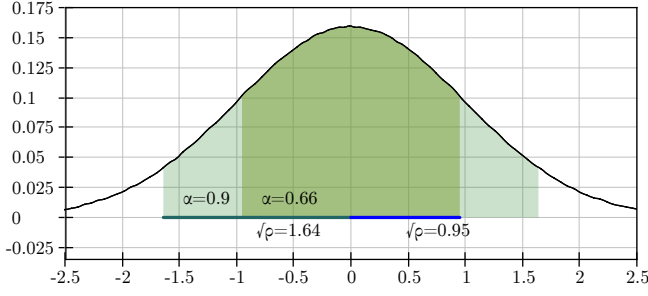
$$\Delta_j^2(\mathbf{x}) = (\mathbf{x} - \mu_j)^T \Sigma_j^{-1} (\mathbf{x} - \mu_j) \quad (11)$$

are  $\chi_D^2$ -distributed, where  $\Sigma_j^{-1}$  is the inverted covariance (or the precision) matrix. With the quantile function  $F_{\chi_D^2}^{-1}$  of the  $\chi_D^2$  distribution, we can determine a squared Mahalanobis distance  $\rho = F_{\chi_D^2}^{-1}(\alpha)$  such that a fraction  $\alpha$  of samples (which belong to the Gaussian) have a smaller squared Mahalanobis distance to the mean as  $\rho$ . Figure 6 depicts the relationship for one- and two-dimensional Gaussians.

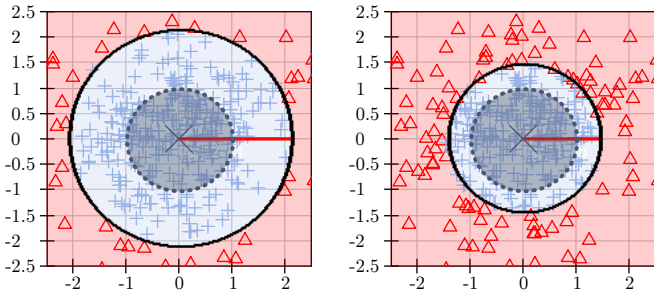
Separating the input space into HDR and LDR simplifies the model considerably. Legitimate samples are assumed to appear in the dense regions while samples in the low-density regions are less likely to be observed. To detect *novel processes* in HDR additional detectors are required (cf. Section 5.2), since 2SND focuses only on LDR. By selecting  $\alpha$  we specify how much of the total probability mass is covered by HDR, thus defining the transition between HDR and LDR. Samples with a Mahalanobis distance of  $\Delta_j(\mathbf{x}) \leq \rho$  to at least one of the component centers  $\mu_j$  are located within a HDR and therefore seen as *not suspicious*. Samples with a higher distance  $\Delta_j(\mathbf{x}) > \rho$  to all centers are regarded as being *suspicious*. This complies to the first stage of our novelty detection.

Figures 6(b) and (c) are illustrate an exemplary situation for a GMM with a single component and different values of  $\alpha$ . Observations inside the  $\alpha$ -region (which is equal to a HDR) are depicted by circles  $\circ$ , while suspicious samples (located in a LDR) are shown as triangles  $\Delta$ . The first stage is implemented in the first part of the procedure 2SND given in Alg. 1.

The second stage utilizes a density based clustering approach. Each sample  $\mathbf{x}'$  that is identified as being suspicious is cached in a circular buffer  $\mathcal{B}$  with size  $b$ . Based on the distance to the nearest neighbor of  $\mathbf{x}'$  in  $\mathcal{B}$ , the algorithm decides if  $\mathbf{x}'$  belongs to an already existing cluster. This behavior depends on the kernel given in Eq. (7) with  $\epsilon$  being the maximum distance between a sample  $\mathbf{x}'$  and its nearest neighbor and it is implemented in the second part of the 2SND procedure, given in Alg. 1. If the sample is associated with a cluster  $C$ , the cluster is extended to include all  $\epsilon$ -neighbors (i. e., all buffered samples with a distance  $\text{dist}(\mathbf{x}', \mathbf{x}_B) \leq \epsilon$ ). This is achieved with the procedure PROPAGATE, which is basically a breadth-first search with constraints. In fact,



(a) A normal distribution with mean  $\mu = 0$  and variance  $\sigma^2 = 1$ . The darker green area shows the region where 66% of the probability mass is located. The combination of both areas corresponds to a mass of 90%. Since the squared distances are  $\chi^2_1$  distributed, the radii of the areas are equal to the root of the quantile function  $F_{\chi^2_D}^{-1}$  of the  $\chi^2_D$  distribution which is  $\sqrt{\rho} \approx 0.95$  for the darker green area (blue line) and  $\sqrt{\rho} \approx 1.64$  for the combined area (green line). The marked areas are identical to the respective high-density regions.



(b) Bivariate Gaussian with 90%  $\alpha$ -region and maximum Mahalanobis distance of  $\sqrt{\rho} = 2.15$ . (c) Bivariate Gaussian with 66%  $\alpha$ -region and maximum Mahalanobis distance of  $\sqrt{\rho} = 1.47$ .

Fig. 6. Relation between normal distribution and  $\chi^2_D$  distributed distances. The dashed ellipses in (b) and (c) are level curves with a Mahalanobis distance of 1 while the black ellipses have a distance of  $\sqrt{\rho} = \sqrt{F_{\chi^2_D}^{-1}(\alpha)}$  to their center. Samples displayed as red triangles  $\triangle$  are *suspicious* (potentially *novel*), while samples depicted as blue circles  $\circ$  are *not suspicious*. High-density regions (HDR) are colored in blue, low-density regions (LDR) in red.

expanding a cluster can lead to a merger of multiple clusters and, thus, create a much larger cluster.

A *novel* process is detected as soon as a cluster  $C$  fulfills the adaptation criterion  $|C| \geq \text{minPts}$  which corresponds to the number of samples that are associated with the cluster  $C$ . In Section 5.3.2 a measure is proposed to represent the current amount of *novelty* in LDR in human readable form.

### 5.1.1 Model Adaptation

The last part of the 2SND procedure is responsible for deciding whether a *novel* process exists in the monitored LDR and how the model is adapted. If a new process in form of a cluster is identified, the underlying GMM needs to be updated. This is done by performing a VI training on all samples that are associated with the corresponding cluster. After that training step, the

### Algorithm 1 2SND

**Input:** sample  $\mathbf{x}'$ , parameters  $\alpha, \epsilon, \text{minPts}, \tilde{b}$   
**Global:** model  $\mathcal{M}$ , buffer  $\mathcal{B}$   
 Initialize  $\rho = F_{\chi^2_D}^{-1}(\alpha)$ .

{1st stage – detection of suspicious samples}

**for all** components  $j$  in  $\mathcal{M}$  **do**

**if**  $\Delta_j^2(\mathbf{x}') \leq \rho$  **then**

    {The observation is not suspicious}

**return** classification of  $\mathbf{x}'$  based on  $\mathcal{M}$ .

**end if**

**end for**

**if**  $|\mathcal{B}| = \tilde{b}$  **then**

  Remove oldest sample from buffer  $\mathcal{B}$ .

**end if**

Add *suspicious* sample  $\mathbf{x}'$  to buffer  $\mathcal{B}$ .

{2nd stage – detection of novel processes}

Find nearest neighbor  $\text{nn}_{\mathbf{x}'}$  of  $\mathbf{x}'$

**if**  $\text{dist}(\mathbf{x}', \text{nn}_{\mathbf{x}'}) \leq \epsilon$  **then**

**if**  $\text{nn}_{\mathbf{x}'}$  belongs to *noise* cluster **then**

    Create new cluster  $C_{\text{new}}$  with samples  $\mathbf{x}'$  and  $\text{nn}_{\mathbf{x}'}$

$C = C_{\text{new}}$

**else**

    Assign  $\mathbf{x}'$  to the same cluster  $C_{\text{nn}}$  as  $\text{nn}_{\mathbf{x}'}$

$C = C_{\text{nn}}$

    PROPAGATE  $C$  to  $\epsilon$ -neighborhood of  $\mathbf{x}'$ .

**end if**

{Process detected – model adaptation}

**if**  $|C| \geq \text{minPts}$  **then**

  Train GMM  $\mathcal{M}_{\text{novel}}$  of process  $C$  with VI.

  Update  $\mathcal{M}$  and fuse it with  $\mathcal{M}_{\text{novel}}$ .

  Remove  $C$  and delete all samples of  $C$  from  $\mathcal{B}$

**return** Classification of  $\mathbf{x}'$  based on updated  $\mathcal{M}$ .

**end if**

**end if**

**return** classification of  $\mathbf{x}'$  based on  $\mathcal{M}$ .

*novel* process is represented by another GMM. To update the model, we exploit the properties of the hyper-distributions and use a fusion technique proposed in [14]. To fuse two given GMM we measure the pairwise divergence between each component and fuse only those which exceed a given threshold (0.5). In this case, we may assume that both components model the same process. This might happen, if a process emerges close to the border if the  $\alpha$ -region (which also separates LDR from HDR). As divergence measure the Hellinger distance is used (cf. [3], [19]). The actual fusion combines the hyper-parameters of both components. Non-overlapping components are simply inserted into the existing model. In each case the hyper-distributions of the mixing coefficients must be adjusted, such that the mixing coefficients of the combined GMM still form a distribution. After this fusion step, the model is adapted to the changes in its

environment and all samples belonging to the cluster  $C$  are removed from buffer  $B$ . If the updated model is used as the base for a classifier, the conclusion for the new component must be determined. Possible solutions to this problem are the involvement of a human domain expert, if meaningful labels are required, or the automatic generation of new, unique labels. As investigated by [12], it is also possible to exchange knowledge with other systems so that a *novel* process can be faster detected by another system. This kind of behavior is especially interesting for cyber-physical systems that share knowledge about their environment.

For clarification, an exemplary scenario that highlights the important operational phases of the new approach is shown in Figure 7.

## 5.2 Novelty Detection in High-Density Regions

The above presented method detects *suspicious* samples in LDR only. However, observations in HDR are always considered as being *normal*, thus making this approach unable to detect *novel processes* there. For the moment we focus on the detection of overlapping processes for single components and later extend the idea to also suit GMM (where multiple components are present). The difficulty in detecting novel processes in dense regions is that we cannot decide if an observed sample is the legitimate outcome of a known (i.e., the existing component) or from an unknown overlapping process without knowing its affiliation (which in this case is a latent variable). We therefore make use of a sliding window that keeps track of the last  $\omega$  observed samples, no matter if they are actually *novel* or not. It is clear, that if a *novel* process is present (that deviates at least in its mean or covariance from the existing component) the observed sample population (i.e., the content of the sliding window) will not match the distribution (described by the component) anymore and a noticeable difference between population and component has to be measurable. Due to their high computational complexity divergence measures such as the Kullback-Leibler divergence [22] or Hellinger distance [19] are intractable for the measurements, especially if the input space is of high dimensionality. To tackle this problem we do not measure the divergence of the sliding window and the existing component directly, instead we test how well the distances between samples in the buffer and the component's center suit the expected distribution (which is a  $\chi^2$  distribution as stated in Section 5.1). This task can be performed by using the  $\chi^2$  test described in Section 4.4. Since the test is performed on the sliding window, the approach is suitable for online environments.

### 5.2.1 Transformation of the distance distribution

One of the requirements of the  $\chi^2$  goodness-of-fit test is, that the different events must be mutually exclusive. Therefore the continuous distance density must be transformed into a discrete one. Since any distribution

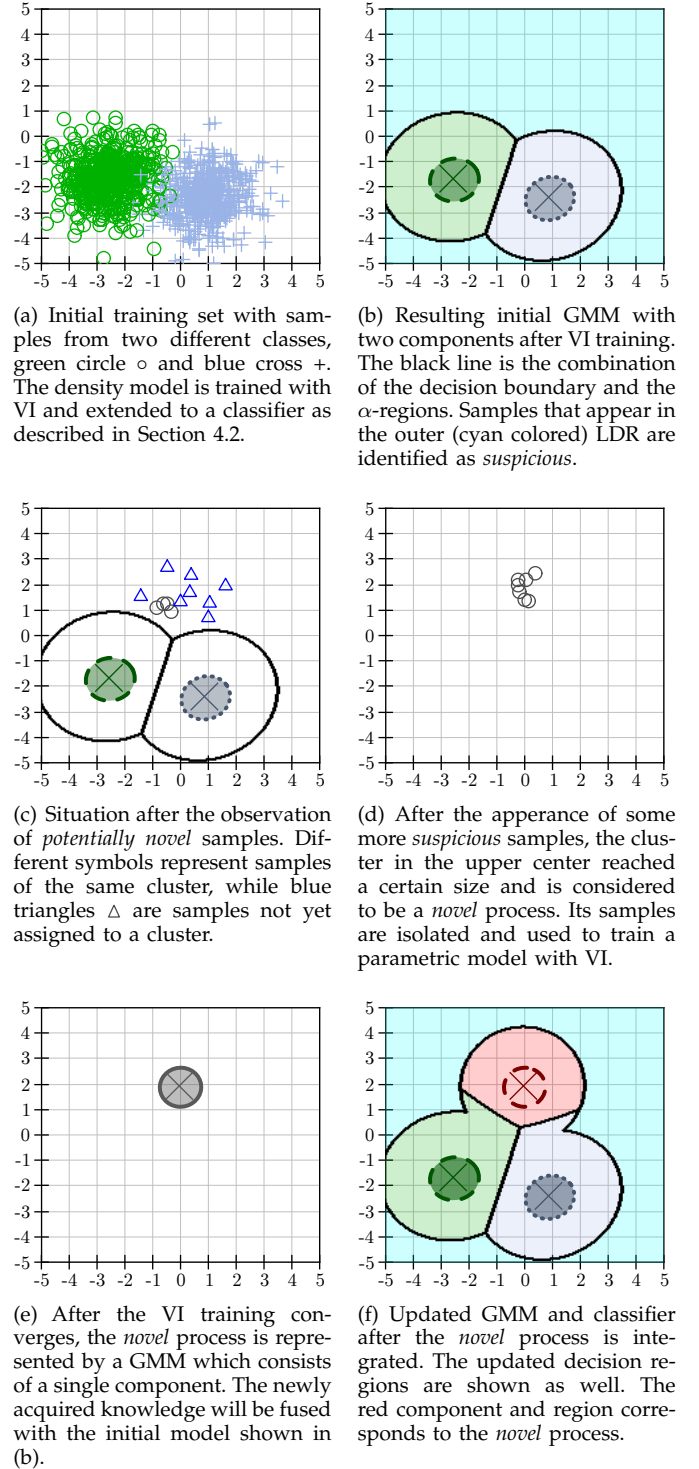


Fig. 7. Illustration of the proposed technique. In the training set only samples of two processes are present. In the operational phase a third process emerges and starts to generate samples. After enough *potentially novel* samples are observed, the model and the classifier are updated.



is normalized (i.e.,  $\int p(x)dx = 1$ ) and, furthermore distances (here  $x$ ) are always strictly positive it is quite easy to transform the density of distances into a discrete, uniform distribution.

Choosing a finite uniform distribution with  $\lambda$  events (also cells, or buckets) brings several advantages including constant calculation of expected frequencies:

$$\text{unif}_\lambda(i) = \frac{1}{\lambda}, \quad 1 \leq i \leq \lambda, i \in \mathbb{N}. \quad (12)$$

The fitted uniform distribution has only one free parameter ( $\lambda$ ), therefore the degree of freedom is given by:

$$k = \lambda - 1. \quad (13)$$

To do the actual transformation the boundaries of the individual cells ( $\text{cell}_i$ ) must be estimated so that each cell is equally likely. This is done with the inverse cumulative density function  $F^{-1}$  of the continuous density (as stated before  $\chi_d^2$  for multivariate Gaussians with  $d$  dimensions) by dividing the density into  $\lambda$  areas of equal size:

$$\text{cell}_i = \begin{cases} [l_i, r_i] & 1 \leq i < \lambda \\ [l_i, \infty) & i = \lambda \end{cases}, \quad (14)$$

$$l_i = F_{\chi_d^2}^{-1} \left( (i-1) \cdot \frac{1}{\lambda} \right), \quad (15)$$

$$r_i = l_{i+1}. \quad (16)$$

Thus the first cell always begins at 0, and the last one is unbounded (right boundary is  $\rightarrow \infty$ ).

Now, if a previously unseen sample  $\mathbf{x}'$  is observed it is stored in the sliding window buffer and a counter  $b_i$  for the responsible cell  $\text{cell}_i$  is incremented. The lookup for the correct cell can be done in  $\mathcal{O}(\log n)$  (e.g., using a tree structure). The  $t$ -value for the current buffer configuration at time point  $n$  is then calculated as follows:

$$t_n = \sum_{i=1}^{\lambda} \frac{(b_{i,n} - e_i)^2}{e_i} \quad (17)$$

with

$$e_i = \text{unif}_\lambda(i) \cdot \sum_{j=1}^{\lambda} b_j \approx \frac{\omega}{\lambda}. \quad (18)$$

The expected value  $e_i$  is only approximated by  $\frac{\omega}{\lambda}$  since the buffer can be not completely filled (i.e., contains less than  $\omega$  samples). If the buffer is at capacity when a new observation  $\mathbf{x}'$  is processed, the oldest element gets removed. The  $t$ -value is compared with significance of  $\alpha = 0.01 = 1\%$  against the critical value:

$$\chi_k^{2,upper} = F_{\chi_k^2}^{-1}(1 - \alpha) = F_{\chi_k^2}^{-1}(0.99). \quad (19)$$

If  $t > \chi_k^{2,upper}$  the threshold is exceeded and the detector reports *novelty*. Figure 8 shows on the left an exemplary data set with a single component trained to fit the green circles  $\circ$ . Additionally the set contains samples from three more processes (purple + crosses around (0.5, 1.5), red  $\triangle$  triangles around (-2, 3), and blue  $\circ$  circles close to the components center around (-2, 1.5)).

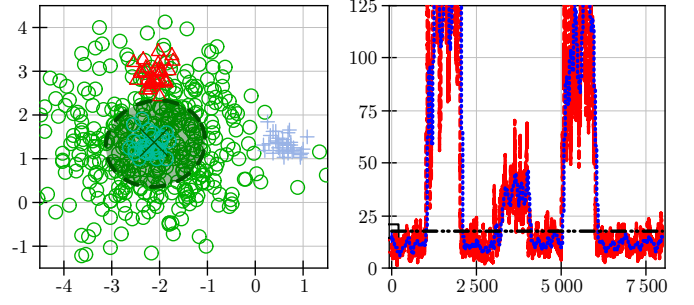


Fig. 8. Test data set and corresponding test statistics  $t$ . First crosses appear, then triangles, and finally circles. The parameters are:  $w = 50$ ,  $\lambda = 12$ ,  $\alpha = 0.99$ ,  $p = 0.01$ . The red line is the test statistics  $t$ . The horizontal, black line indicates the critical value.

All additional processes represent *novelties* and appear in the given order. The image on the right shows the curve of the calculated  $t$ -values for the sliding window in red. When samples from the *novel* processes appear the curve changes and exceeds the critical value (given as black line) considerably.

The signal is however noisy, so that at some points of time the critical value is slightly exceeded even though no *novel* processes are present. To compensate for this effect smoothing the  $t$ -values with a moving average:

$$t_{ma,n} = \frac{1}{M} \sum_{i=0}^M t_{n-i}, \quad (20)$$

where  $M$  is the number of considered previous  $t$ -values, is a promising approach as the blue curve on the right image of Figure 8 illustrates.

### 5.2.2 Learning of the distance distribution

Real world data sets or sensory data from embedded systems often differ from their assumed distributions. Whereas this is not a problem for classification for our goodness-of-fit approach to *novelty detection* it is, as the  $t$ -value curve in Figure 9 highlights. Here a single Gaussian is fitted based on the depicted samples, which are uniformly distributed rather than normal. Therefore the distances are not  $\chi^2$  distributed as it is presumed by our test and the critical value is almost permanently exceeded by the test statistics.

The problem can be solved by estimating the cell boundaries directly from the samples  $\mathcal{X}_{train}$  that are used to train the component:

$$\text{cell}_i = \begin{cases} [l_i, r_i] & 1 < i < \lambda \\ [0, r_i] & i = 1 \\ [l_i, \infty) & i = \lambda \end{cases}, \quad (21)$$

$$l_i = \Delta(\mathbf{x}_{\lceil i \cdot \frac{w}{\lambda} \rceil}), \quad \mathbf{x}_j \in \text{Sort}(\mathcal{X}_{train}), \quad (22)$$

$$r_i = l_{i+1}. \quad (23)$$

where  $\Delta(\mathbf{x})$  is the Mahalanobis distance to the center  $\mu$  of the component. By using the distance of every  $\lceil i \cdot \frac{w}{\lambda} \rceil$

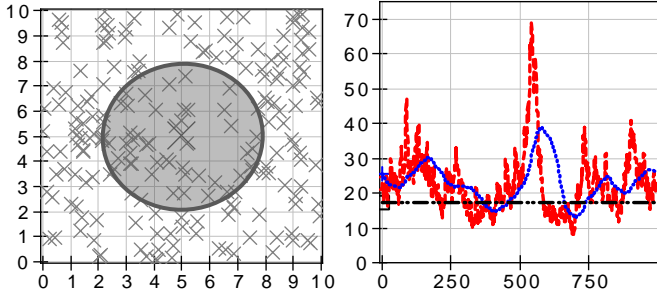


Fig. 9. 1000 Uniformly distributed 2D samples in the interval  $[0, 10]$  and *trained* Gaussian component. On the right the test statistics (red) and moving average test statistic (blue) of the whole applied train set. The critical value (black line) is clearly exceeded.

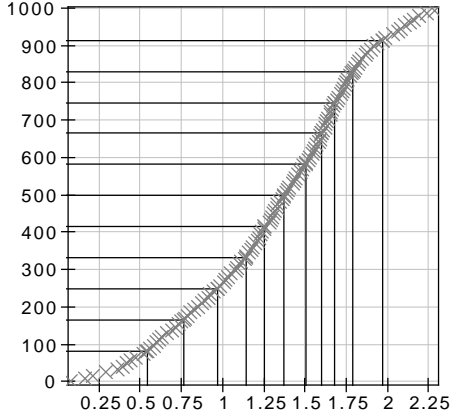


Fig. 10. Ordered mahalanobis distances to the center of observed samples drawn from a bounded uniform distribution ( $x \in [(x_1, x_2) | x_1, x_2 \in \text{unif}(0, 100)]$ ). The horizontal and vertical lines mark the boundaries for the cells of the resulting discrete uniform distribution.

element of the order samples each cell contains approximately the same number of entries, thus forming again a discrete uniform distribution. The computationally most complex part is the sorting of the training samples  $\mathcal{X}_{train}$ , which can be done in  $\mathcal{O}(n \cdot \log n)$ . Note that the last cell  $\lambda$  might be underestimated due to the rounding.

In Figure 10 the ordered training set  $\mathcal{X}_{train}$  is depicted. The  $y$ -axis (index of the sorted samples) is divided into  $\lambda = 12$  equally sized parts (each part corresponding to a cell) the associated function arguments on the  $x$ -axis (distances to component center) are equal to the interval boundaries. If the curve is normalized an approximation of the cumulative density function of the real distance distribution can be obtained.

Figure 11 shows the resulting  $t$ -value curves of two uniform distributed data sets (on the left the same example as in Figure 9 with 2 dimensions, on the right another sample set with 5 dimensions) where the expected frequencies for the test are estimated according to Equation (21). The (moving-average) curves are now clearly below

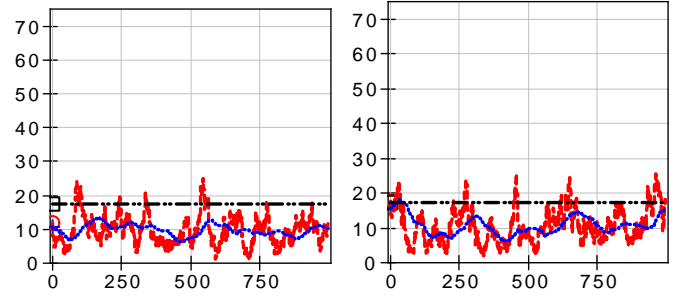


Fig. 11. Test statistics (red) and moving average test statistic (blue) each for uniformly distributed samples with estimated distance distribution. On the left side for 1000 two dimensional samples, on the right for 1000 five dimensional samples.  $w = 50$ ,  $\lambda = 12$ ,  $ma = 100$ ,  $p = 0.01$

the critical value (black line), thus not indicating any *novelty*.

### 5.2.3 Extension to Gaussian Mixture Models

To extend the high-density approach to GMM with multiple components, each component needs its own detector.

If a new sample  $\mathbf{x}'$  is observed it should be used to update the detector of its affiliated component. The affiliation is however a latent variable and thus not known at run-time. One method to estimate the affiliations is (Monte Carlo) random sampling, which requires only the evaluation of the unnormalized (without mixing coefficient  $\pi_j$ ) densities  $P_j(\mathbf{x}') = \mathcal{N}(\mathbf{x}' | \mu_j, \Sigma_j)$  for each component  $j$  and a continuous uniform pseudo random number generator  $\text{unif}(0, 1)$  for the unit interval  $[0, 1]$ . The sampling works by partitioning the unit interval into  $J$  parts. Each partition  $m_j$  is associated with exactly one component  $j$  and the boundaries are given by:

$$m_j = \begin{cases} [0, r_j) & j = 1 \\ [l_j, r_j) & 1 < j < J, \\ [l_j, 1] & j = J \end{cases} \quad (24)$$

$$p_{\mathbf{x}', j} = \frac{P_j(\mathbf{x}')}{\sum_{k=1}^J P_k(\mathbf{x}')} \quad (25)$$

$$l_j = r_{j-1} \quad j > 1, \quad (26)$$

$$r_j = l_j + p_{\mathbf{x}', j}. \quad (27)$$

where  $p_{\mathbf{x}', j}$  are the normalized densities to ensure that the support of the individual parts sum up to 1. To identify a *winner* component (i.e., the one that will be affiliated with the observation  $\mathbf{x}'$ ) a random value  $r'$  is drawn from the uniform generator  $\text{unif}(0, 1)$ . The partition  $m_j$  that covers the drawn value  $r'$  indicates the *winning* component  $j$ .

Figure 12(a) shows *clouds* (5000 samples, 2 dimensions, 2 classes) a widely used artificial data set from the UCI Repository [23], with trained classifier. The CMM is trained in 5-fold cross-validation fashion, where four

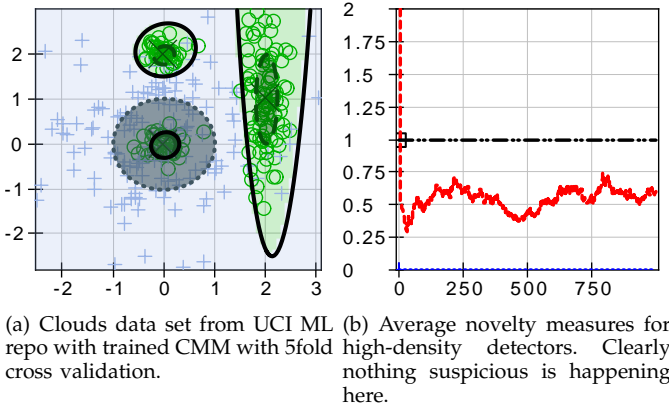


Fig. 12. Data set with trained CMM and corresponding *average-novelty* curve.

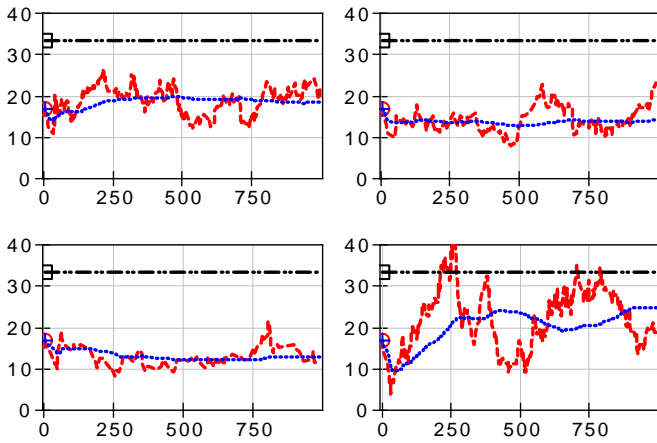


Fig. 13. Test statistics (red) and moving average test statistic (blue) each for remaining folds applied to all detectors.  $w = 50$ ,  $\lambda = 12$ ,  $ma = 100$ ,  $p = 0.01$

folds are used for the actual training and the remaining fold for testing. This leads to an experiment where no *novel* (unknown) processes are present since all folds contain samples from all four known processes. The test result for one experiment is displayed in Figure 12(b). The curve shows the average high-density novelty measure (discussed in Section 5.3.2), which does not exceeded the critical value that is given by the black line and has a constant value of 1, thus indicating that no *novel* processes are present. The test statistics of the individual component detectors are depicted in Figure 13.

A modified *test* setup for clouds is illustrated in Figure 14(a). Here the training is performed on 2000 samples and the remaining 3000 samples are interspersed with 400 samples from two overlapping *novel* processes (red  $\triangle$  triangles). The novel processes appear around time steps 1000 and 2200. The corresponding high-density novelty measure curve for the experiment is given in Figure 14(b) and indicates *novelty* (blue bars rising to 1) in the regions where the *novel* samples are interspersed. The test statistics of the individual component detectors

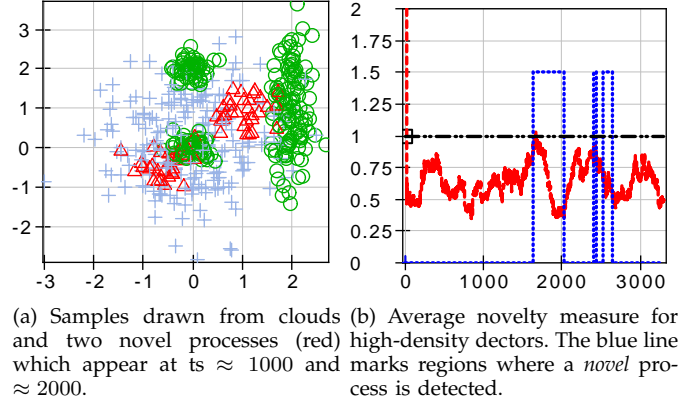


Fig. 14. Data set with interspersed samples from *novel* processes and corresponding *average-novelty* curve.

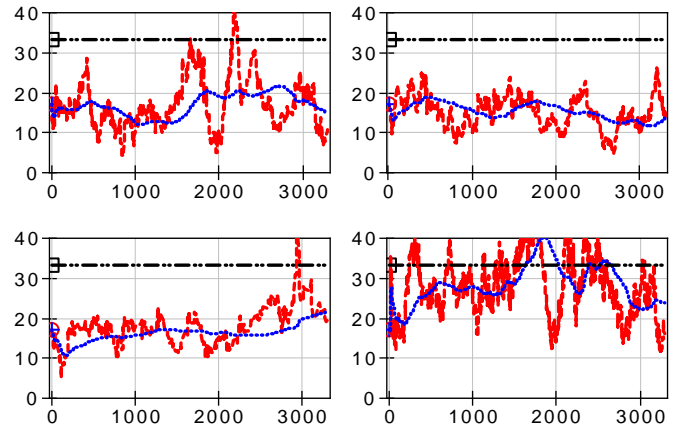


Fig. 15. Test statistics (red) and moving average test statistic (blue) each for modified clouds applied to all detectors.  $w = 50$ ,  $\lambda = 12$ ,  $ma = 100$ ,  $p = 0.01$

of this experiment are depicted in Figure 15. From the curves it can be inferred that the main contributions for the detection is from the component (bottom right) that represents blue + crosses.

### 5.3 CANDIES

CANDIES is our holistic approach to detect novelty in high- as well as in low-density regions of a GMM. This is achieved by using a single 2SND (cf. Section 5.1) detector combined with multiple HDR detectors for each component (cf. Section 5.2.3).

#### 5.3.1 Requirements to merge LDR and HDR detectors

For the system to operate some adjustments are necessary. At first a previously unseen sample  $x'$  is checked whether it is located in an HDR or not (similar to the first stage of 2SND). If this is not the case, the sample is passed to the second stage of 2SND and the density based clustering is refreshed. At this point a *novel* process might be detected. Otherwise, the in Section 5.2.3 described random sampling is executed and  $x'$  gets affiliated with exactly one of the components  $J$ . Then

---

**Algorithm 2** CANDIES
 

---

**Input:** sample  $\mathbf{x}'$ , parameters  $\alpha, \omega, \epsilon, \minPts, \tilde{b}$   
**Global:** model  $\mathcal{M}$ , buffer  $\mathcal{B}$   
 Initialize  $\rho = F_{\chi_D^2}^{-1}(\alpha)$ .

{decide if  $\mathbf{x}'$  is located in high- or low-density region}

**for all** components  $j$  in  $\mathcal{M}$  **do**

**if**  $\Delta_j^2(\mathbf{x}') \leq \rho$  **then**

    {The observation is in dense region}

    {get winner according to Section 5.2.3}

$j$  = winner component for  $\mathbf{x}'$

    update  $\chi^2$  detector of component  $j$

    {Compare  $t$ -value with critical value}

**if**  $t_j > \chi^{2,upper}$  **then**

      {Process detected}

**end if**

**return** classification of  $\mathbf{x}'$  based on  $\mathcal{M}$ .

**end if**

**end for**

{The observation is in low-density region}

**return** 2SND( $\mathbf{x}', \alpha, \epsilon, \minPts, \tilde{b}$ ).

---

for this component  $j$  a new  $t$ -value is estimated and compared against the critical value. At this point an overlapping *novel* process might be detected.

Since samples  $\mathbf{x}'$  with a distance  $\Delta_j(\mathbf{x}') > \rho$  for all  $j \in J$  are always processed by the LDR detection part, the assumed distance distribution of the components HDR detectors will not match the observed samples. However, by establishing the following dependency  $\lambda = \frac{1}{1-\alpha}$  between the 2SND LDR detector and the HDR detectors, and adjusting the calculation of the  $t$ -values to:

$$t_{n,j} = \sum_{i=1}^{\lambda-2} \frac{(b_{i,n} - e_i)^2}{e_i}, \quad (28)$$

$$e_i \approx \frac{\omega}{\lambda - 1}, \quad (29)$$

the last cells  $\text{cell}_\lambda$  are representing exactly the fraction  $\alpha$  of samples that are located in LDR (but with  $e_\lambda = 0$ ) while the first  $\lambda - 1$  cells cover the remaining  $1 - \alpha$  percentage. The critical value is changed to:

$$\chi_{\lambda-2}^{2,upper} = F_{\chi_{\lambda-2}^2}^{-1}(0.99). \quad (30)$$

Thus the goodness-of-fit test is adjusted to evaluate only the frequencies of samples expected to appear in HDR. The whole approach is summarized and commented in Algorithm 2.

Figure 16 shows another modification of the previously used *clouds* data set. Again, additional samples (red  $\triangle$  triangles) from two *novel* processes are interspersed. The locations are chosen in a way so that one process (centered around (1, 1)) shares a large fraction of its support with two of the known processes, while the other one (centered around (-2, -2)) is positioned in a

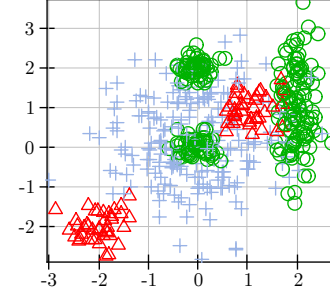


Fig. 16. Samples drawn from clouds and two novel processes (red) which appear at  $ts \approx 1000$  and  $\approx 2000$ .

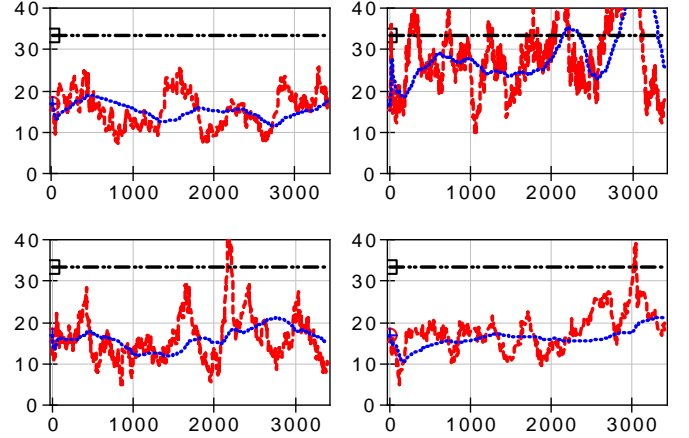


Fig. 17. Test statistics (red) and moving average test statistic (blue) each for modified clouds applied to all detectors.

low-density region. The  $t$ -value curves of the four *known* components are illustrated in Figure 17. Furthermore the *novelty measures* (discussed in Section 5.3.2) given in Figure 18 clearly indicate *novelty* in the expected time ranges (blue bars rising to 1). The left curve gives the novelty value in low-density regions, where the first *novel* process starts generating samples around time stamp  $\approx 1000$ . On the right curve the *average novelty* measure for high-density regions is illustrated. Here, the *novel* process gets also detected but a small delay between appearance of *novel* samples (around  $ts \approx 2000$ ) and the detection can be observed. This is most likely due to the *random sampling*, which disperses *novel* samples to multiple components.

### 5.3.2 Novelty Measure – Human Readability

We propose two *novelty measures* to quantify how much *novelty* is present in different regions (i.e. HDR or LDR) in a way that is comprehensible for (data scientists). Therefore, the measures should express the absence of a *novel* process (or *novelty*) with a value near 0, while the presence of such a process should be expressed by a value  $\geq 1$ .

The measure  $\nu_{2snd}$  for LDR is given by:

$$\nu_{2snd,n} = 1 - \frac{|C| + |Noise|}{|B|} \quad (31)$$



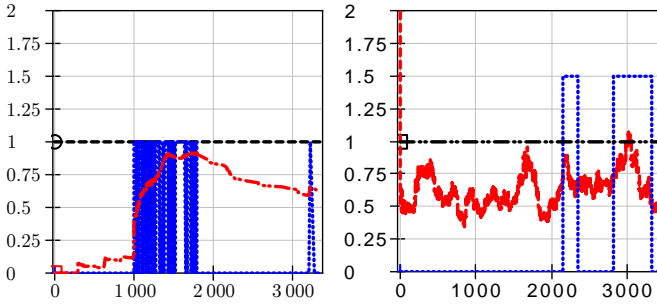


Fig. 18. Measured novelty for modified clouds (cf. Figure 16). On the left: *2SND-novelty* curve indicating a *novel* process in low-density regions around  $ts \approx 1000$ . On the right: *average-novelty* curve that represents *novelty* in high-density regions. The rising of the blue bars around  $ts \approx 2150$  and  $\approx 2900$  are indicating that a *novel* process is present.

Where  $|\mathcal{B}|$  is the number of observations currently stored in the buffer,  $|\mathcal{C}|$  is the number of different cluster, and  $|\text{Noise}|$  the number of samples associated with the *noise* cluster. If a single cluster that contains most samples currently kept in the buffer is present (which is a strong indicator for a novel process), the measure will be close to 1. On the other hand, if all samples are considered to be *noise* or multiple clusters with only a few samples are present, the *novelty* value will be closer to 0.

The HDR measure  $\bar{\nu}$  (*average novelty*) is based on the geometric mean of the normalized  $t$ -values  $\nu_j$  of the individual components:

$$\bar{\nu}_n = \left( \prod_j^J w(\nu_j) \right)^{\frac{1}{J}}, \quad (32)$$

$$\nu_j = \frac{t_{n,j}}{\chi_{k,2,upper}^2}. \quad (33)$$

The normalization constant is given by the critical value. As Equation (32) shows, the  $\nu_j$  are passed to a function  $w$  which is a non-linear transform that boosts values near 1:

$$w(x) = x \cdot (2 - \text{comp}(1 - x, 1000)), \quad (34)$$

$$\text{comp}(x, \mu) = \frac{\log(1 + \mu \cdot x)}{\log(1 + \mu)}. \quad (35)$$

The idea here is that if multiple components approach the critical value (an indicator for *novel* process located between these components) the *novelty measure* should also express this. If the model however consists of considerably more components (with  $t$ -values distant from the critical value), the mean is dominated by these components. Thus boosting values already close to 1, allows to overcome the *normal* components to increase the mean, so that *novelty* is also expressed there. Exemplary curves for both measures are given in Figure 18 (*novel* processes present) and Figure 19 (only *normal* processes observed).

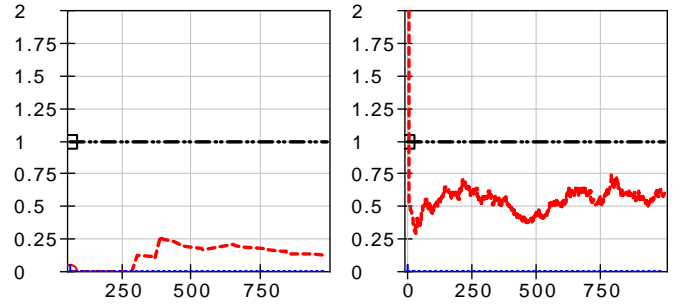


Fig. 19. Measured novelty for clouds. On the left: *2SND-novelty* curve showing low *novelty* values for low-density regions. On the right: *average-novelty* curve that represents *novelty* in high-density regions.

### 5.3.3 Overview of Parameters

CANDIES comes along with a considerable amount of adjustable parameters. Table 1 gives an overview of all parameters present in CANDIES including a short description, recommendations for (good) default values (if possible), and which detector is influenced by the parameter. Note, that especially the buffer-size parameters are application-dependent on how many *novel* processes are expected to appear at once, and how many samples they will generate.

### 5.3.4 Handling of Noise

Since the novelty detection is designed to detect *novel* processes and not single observations, it is rather robust against distributed noise in the input space. While *novel* samples of a *novel* process will appear in a dense form, random noise is scattered across the input space so that it is quite unlikely to form sufficiently large clusters.

For the LDR detection part (based on 2SND) the robustness is achieved by the two stage architecture that *suspicious* samples pass through. Figure 20 shows a scenario that includes uniformly distributed noise that is mixed into a test set with observations from one known process and one novel process (located to the right, outside of the  $\alpha$ -zone in low-density region). Depending on its parametrization, the LDR approach only detects a *novel* process where the *novel* observations are actually located.

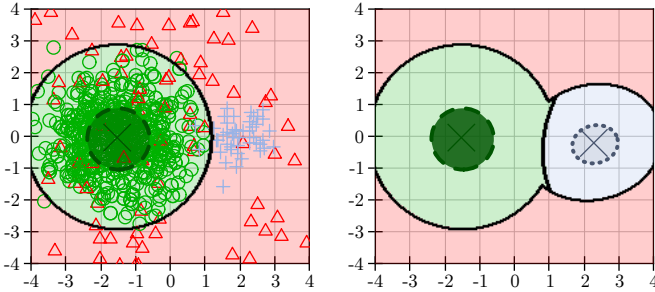
Figure 21(a) depicts the same exemplary data set that is already used in Section 5.2.1 interspersed with uniform random noise (purple + crosses). The corresponding  $t$ -value curve is displayed in Figure 21(b) and shows a recognizable up-shift, introduced by the noise. Nevertheless, this undesired effect can be circumvented by adjusting the distance distribution according to Section 5.2.2. The curve of the adjusted test is illustrated in Figure 21(c). The course of the *moving-average* is now clearly below the threshold in intervals where no *novelty* is present, but rises clearly - although weaker as compared to the application without noise - above the critical value, when the *novel* processes start to generate samples. Therefore the high-density approach



TABLE 1  
Different parameters necessary for the proposed combined novelty detection approach.

Parameter	Description	Default	Detector
$\omega$	Window size per component detector	$5 \cdot \lambda$	High-Density
$\lambda$	Number of discrete levels	$\frac{1}{1-\alpha}$	High-Density
ma	Size of MovingAverage filter	$2 \cdot \omega$	High-Density
p	$\alpha$ -Value for $\chi^2$ test	0.01	High-Density
$\alpha$	Size of alpha region	0.95	Low-Density: 1. Stage
$\epsilon$	Maximum distance between samples in a cluster	2	Low-Density: 2. Stage
$ \mathcal{B} $	Buffer for Samples in low density region	100	Low-Density: 2. Stage
$P(\mathcal{C})=\text{minPts}$	Size of a cluster to be considered as the outcome of a new process	10	Low-Density: 2. Stage

is essentially capable of handling noise, but requires the presence of noise in the training data.



(a) Test set consisting of samples from one previously known process (green circles  $\circ$ ), noise samples (red triangles  $\Delta$ ) and samples from a novel process (blue crosses  $+$ ).  
(b) Model identified by our approach after the process (blue  $\times$  surrounded by dotted ellipse) that was responsible for the blue crosses is detected and integrated into the classification model.

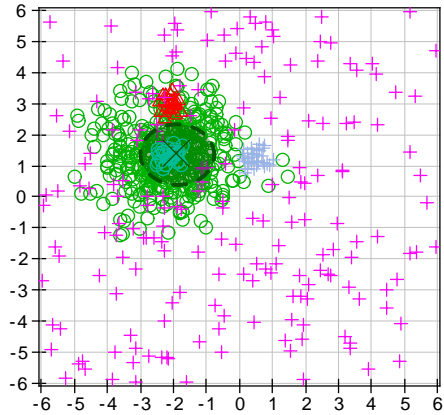
Fig. 20. Scenario with samples from a *novel* process and additional uniform distributed noise that is scattered in the input space. The region where observations are identified as *novel* is colored in red, regions with different class assignments are also separated by a solid black decision boundary.

### 5.3.5 An alternative view on CANDIES

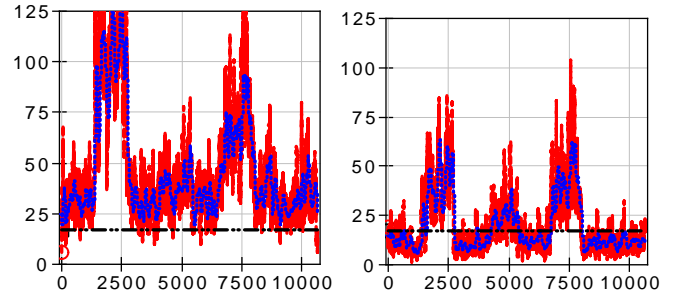
At first the two different approaches to novelty detection for LDR and HDR might seem quite different. It is, however, possible to get a consistent view by interpreting one detector by means of the other. As mentioned before, the last bins  $\text{cell}_\lambda$  of each HDR detector matches the low-density parts of the input space. Therefore the ring buffer  $\mathcal{B}$  used for 2SND can be seen as a shared *cell* across all HDR detectors. On the other hand, the individual buffers of each HDR detector allow an interpretation as *clusters* (with a different adaptation predicate  $P$ ), and thus suiting the 2nd stage of 2SND.

## 6 CASE STUDY

To validate that the presented approach can be used to real-world applications, we show experimental results



(a) The same data set as in Figure 8 but with uniform noise added (purple crosses  $+$ ).



(b) Test statistics over time for non-adjusted test. (c) Test statistics over time for estimated distance distribution.

Fig. 21. Scenario with samples from three *novel* processes and additional uniform distributed noise that is scattered in the input space and corresponding test curves: test statistics (red) and moving average test statistic (blue).

based on the well-known *KDD Cup 1999* network intrusion data set [21]. Even though it is pointed out that there are some serious flaws in the data set, which makes it inappropriate for the evaluation of real intrusion detection systems, its properties are still suitable for our purposes, since we are not interested in building a state of the art intrusion detection system.

## 6.1 Setup

As mentioned before, our new approach is compared to a novelty detection technique that is proposed in [12]. Here, *novel* samples are also identified using a GMM and the squared Mahalanobis distance between processed samples and the mean of the different components. Each time a new sample is processed, an internal state variable  $S_n$  is updated, such that  $S_n = S_{n-1} + \chi_{nov}^2$ , with:

$$\chi_{nov}^2(\mathbf{x}) = \eta \sum_{j=1}^J p(j|\mathbf{x}) \left( \delta_{\alpha,j}(\mathbf{x}) - \frac{\alpha}{1-\alpha} (1 - \delta_{\alpha,j}(\mathbf{x})) \right) \quad (36)$$

being a penalty or reward, depending on how well the new sample fits the model. To compute whether the state variable is rewarded or punished, the indicator functions:

$$\delta_{\alpha,j}(\mathbf{x}) = \begin{cases} 1, & \Delta_j^2(\mathbf{x}) \leq \rho = F_{\chi_D^2}^{-1}(\alpha) \\ 0, & \text{sonst} \end{cases} \quad (37)$$

of each component are evaluated and the results are multiplied with the responsibilities of the components. If the algorithm is processed in an environment without emerging processes, the expectation of the state variable will be equal to its initial value  $E[S_n] = 1$ . The presence of a *novel* process will lead to a decrease of the value of the state variable  $S_n$ . This can be exploited to detect *novel* processes as soon as the state variable underflows a given threshold (here: 0.2). The parameter  $\eta$  controls how fast the state variable changes (here: 0.001). This causes a model adaptation that uses the last 500 observations to retrain the model, which is done with a modified VI algorithm, that allows to insert new components into an existing GMM and train only those, keeping the existing components “fixed”. After the model is adapted to its changed environment, the state variable is reset to its initial value. We refer to this approach as **CSND** ( $\chi^2$ -novelty detection)

Originating from the various recorded connections in the KDD99 data set, different attack *scenarios* are sampled (these are: ipsweep, neptune, nmap, satan, and smurf). Each scenario consist of background connections (legitimate network traffic) and connections related to the specific attack. A dimension reduction to 6 out of the 41 dimensions is performed as preprocessing step. Additionally and due to the massive support in terms of categories, we interpret the discrete attributes as nearly continuous. Each scenario consists of three parts with an overall of 25000 connections. The first part contains 10000 connections drawn from a pool of background connections only. The second part is a mixture of background and attack connections (with the attack name as label) with a ratio of 3:1 and a total of 10000 connections. The last 5000 connections form the third part, which again consists only of legitimate traffic.

Both *adaptive* classifiers are trained with the first 5000 samples of the first part of each scenario to learn an initial GMM with VI. The experiments themselves are

conducted in a 5-fold cross-validation fashion, with independent folds for the *train sets*, which consist of connections from the first and third parts of each scenario, and a single *test set* that is equal to the second part of each scenario.

Additionally, to get a baseline for the classification performance, a static classifier (as described in Section 4.2, referred to as GMM-Static) is trained on samples of all classes (background connections and attacks). That is, this classifier can be seen as omniscient as it anticipates future attacks that are completely new and unpredictable for the two adaptive classifiers above. In order to get meaningful results, a stratified 5-fold cross-validation, with all connections mixed together, is carried out. Then the accuracy and the  $F_1$ -score of the *class* assigned to samples of the *novel process* are used to evaluate the classification performance.

### 6.1.1 Results

The resulting averaged classification performances are summarized in Table 2, which states that both adaptive approaches are able to identify the attacks and perform model adaptations that integrate the acquired knowledge. In all scenarios, the accuracy and the observed  $F_1$ -score of CANDIES is equal or higher compared to those of the CSND approach. In three out of five scenarios our approach performs comparably well as the static baseline and still satisfiable on the other two.

TABLE 2

Comparison of classification accuracies and  $F_1$ -scores for the *novel process* (in form of the applied attack) of both novelty detection approaches and the GMM-Static baseline.

SCENARIO	CANDIES		CSND		GMM-STATIC	
	ACC	$F_{1nov}$	ACC	$F_{1nov}$	ACC	$F_{1nov}$
IPSWEET	97.1%	(0.9)	86.1%	(0.1)	96.2%	(0.9)
NEPTUNE	99.3%	(1.0)	94.7%	(0.8)	98.8%	(1.0)
NMAP	95.3%	(0.8)	90.9%	(0.4)	98.0%	(0.9)
SATAN	95.3%	(0.7)	92.3%	(0.7)	99.0%	(1.0)
SMURF	99.2%	(1.0)	92.8%	(0.7)	99.8%	(1.0)
$\emptyset$	97.2%	(0.9)	91.4%	(0.5)	98.4%	(0.9)

The higher performance of CANDIES over CSND is explained by Table 3, which shows the average number of *actual novel* samples (samples actually belonging to the attack) that are processed before the *novel process* is detected and a model adaptation triggered. Here CANDIES displays its strength to exploit spatial information between *suspicious* samples in LDR the form of clusters, which accelerates the detection compared to the slowly changing state variable of CSND.

The algorithm is designed to be processed in an online mode. Therefore, the number of triggered model adaptation steps and the number of inserted components are also investigated. Table 4 shows the averaged number of adaptation and insertion steps for each scenario. As

TABLE 3

Number of actual *novel* samples needed until a *novel* process gets detected and a model adaptation is triggered.

SCENARIO	CANDIES	CSND
	REQUIRED OBSERVATIONS	
IPSWEET	40.4	1063.4
NEPTUNE	50.6	1043.0
NMAP	85.0	1173.2
SATAN	19.4	1151.8
SMURF	37.2	1041.4
$\emptyset$	46.5	1094.6

TABLE 4

Number of triggered model adaptations and average number of inserted components (in parentheses) for both approaches.

SCENARIO	CANDIES		CSND	
	ADAPT.	COMP.	ADAPT.	COMP.
IPSWEET	1.0	(3.0)	2.0	(13.6)
NEPTUNE	1.0	(1.8)	1.0	(1.0)
NMAP	1.2	(3.2)	1.0	(5.8)
SATAN	2.0	(5.0)	1.0	(4.8)
SMURF	1.6	(3.0)	1.0	(7.8)
$\emptyset$	1.4	(3.2)	1.2	(6.6)

we can see, both approaches tend only to a single model adaptation, which is the optimum here. The CSND approach has fewer model adaptations on average than the CANDIES, but has a higher average number of inserted components, which is not negligible since the number of components in the GMM has a direct influence on the run-time of both algorithms.

## 7 CONCLUSION AND OUTLOOK

We introduced CANDIES, a holistic approach to novelty detection for (new) emerging processes throughout the complete input space of a probabilistic classifier. To achieve this, different novelty detectors for *low-density* regions (LDR, where it is less likely to observe samples) and *high-density* regions (HDR, samples are expected to be observed here) are combined and thus able to cover the complete input space. For LDR we resort on 2SND, this algorithm works with two stages. First, *suspicious* observations are identified with the help of a GMM (which are based on parametric densities). In the second stage, *suspicious* samples are then clustered in a nonparametric way (inspired by DBSCAN). A *novel* process is recognized as soon as one of the (nonparametric) clusters reaches a sufficient size. The detection in HDR on the other hand is purely based on *parametric density estimation*. We showed how to use multiple detectors (one detector per component) to identify *novelty* in GMM. The presence of *novel processes* in HDR is

directly identified. This is accomplished by maintaining a *sliding window* of recent observations and performing statistical *goodness-of-fit* tests between *sliding window* and the affiliated component.

In a compact case study in the field of computer network intrusion detection, we could show that CANDIES is applicable to real-world data sets. We tested it on a subset of the well-known KDD Cup '99 Intrusion Detection data set, where rather promising results were obtained. So far, first experiments on artificial laboratory data sets lead us to the conclusion that CANDIES will be a satisfactory solution to *novelty detection* with model adaptation in the near future.

In our future work we will focus on extending the described *novelty* detector further, this includes in particular *reaction* procedures for the HDR detection. We will elaborate the performance of CANDIES on more sample applications, e.g., in the fields of robotics or video based surveillance. Detection and handling of obsolescence or concept shift will be accomplished with techniques similar to the ones proposed here. The same holds for concept drift, but here, it will be quite difficult to effect the trade-off between under- and overreaction (too early or too late). The accuracy of our techniques must be set in relation to a "degree" of time-variance in the observed system. It will be possible to detect emergent phenomena in the observed environment and to numerically assess the degree of emergence (cf. [13]). Also, these techniques allow for an application to various anomaly detection problems. Furthermore, the design of the approach is not necessarily limited to GMM but applicable to other mixture models as well.

Another possible application field that could benefit from our proposed technique are systems equipped with *awareness* capabilities. Often, terms such as location-aware, context-aware, self-aware, or environment-aware are used in the literature (see, e.g., [1], [26]). In our opinion, awareness is essentially

- the capability to compare knowledge about the self, the environment, other systems etc. to current observations in order to detect when expectations concerning current observations do not meet the actual observations anymore and
- the ability to adapt the knowledge model in a way such that the system meets some performance requirements which includes a solution to the problem when to adapt the model in order to avoid a performance loss either due to too fast or too slow reactions.

Altogether, awareness techniques will be a key to develop new kinds of technical systems that could actually be termed to be "intelligent" or "smart" with some higher degree of justification.

## ACKNOWLEDGMENTS

The authors would like to thank the German Research Foundation (DFG) for support within the DFG project CYPHOC (SI 674/9-1).

## REFERENCES

- [1] G. Abowd, A. Dey, P. Brown, N. Davies, M. Smith, and P. Steggles. Towards a better understanding of context and context-awareness. In H.-W. Gellersen, editor, *Handheld and Ubiquitous Computing*, volume 1707 of *Lecture Notes in Computer Science*, pages 304–307. Springer Berlin Heidelberg, 1999.
- [2] H. Al-Behadili, A. Grumpe, C. Dopp, and C. Wohler. Extreme Learning Machine based Novelty Detection for Incremental Semi-Supervised Learning. In *International Conference on Image Information Processing*, pages 230–235, 2015.
- [3] A. Bhattacharyya. On a measure of divergence between two multinomial populations. *Sankhyā: The Indian Journal of Statistics*, pages 401–406, 1946.
- [4] C. M. Bishop. Novelty detection and neural network validation. In *Vision, Image and Signal Processing*, volume 141, pages 217–222, 1994.
- [5] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, 2006.
- [6] J. Bonifacio, A. Cansian, A. Carvalho, and E. Moreira. Neural networks applied in intrusion detection systems. In *Proc. of IJCNN*, volume 1, pages 205–210, 1998.
- [7] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. LOF: Identifying Density-Based Local Outliers. *ACM SIGMOD Record*, 29(2):93–104, 2000.
- [8] D. A. Clifton, S. Huguency, and L. Tarassenko. Novelty detection with multivariate extreme value statistics. *Journal of Signal Processing Systems*, 65(3):371–389, 2011.
- [9] L. Clifton, D. A. Clifton, P. J. Watkinson, and L. Tarassenko. Identification of patient deterioration in vital-sign data using one-class support vector machines. *2011 Federated Conference on Computer Science and Information Systems (FedCSIS)*, (2):125–131, 2011.
- [10] L. Clifton, D. A. Clifton, Y. Zhang, P. Watkinson, L. Tarassenko, and H. Yin. Probabilistic novelty detection with support vector machines. *IEEE Transactions on Reliability*, 63(2):455–467, 2014.
- [11] M. Ester, H. Kriegel, S. J., and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. of KDD-96*, pages 226–231. AAAI Press, 1996.
- [12] D. Fisch, M. Jänicke, E. Kalkowski, and B. Sick. Techniques for knowledge acquisition in dynamically changing environments. *TAAS*, 7(1):1–25, 2012.
- [13] D. Fisch, M. Jänicke, B. Sick, and C. Müller-Schloer. Quantitative emergence – a refined approach based on divergence measures. In *SASO*, pages 94–103, 2010.
- [14] D. Fisch, E. Kalkowski, and B. Sick. Knowledge fusion for probabilistic generative classifiers with data mining applications. *TKDE*, 26(3):652–666, 2014.
- [15] C. Gruhl, B. Sick, A. Wacker, S. Tomforde, and J. Hähner. A building block for awareness in technical systems: Online novelty detection and reaction with an application in intrusion detection. In *IEEE iCAST*, pages 194–200. IEEE, 2015.
- [16] J. Haehner, U. Brinkschulte, P. Lukowicz, S. Mostaghim, B. Sick, and S. Tomforde. Runtime Self-Integration as Key Challenge for Mastering Interwoven Systems. pages 1–8, 2015.
- [17] V. Hautamäki, I. Kärkkäinen, and P. Fränti. Outlier detection using k-nearest neighbour graph. *Proc. – International Conference on Pattern Recognition*, 3(09):430–433, 2004.
- [18] A. Hazan, J. Lacaille, and K. Madani. Extreme value statistics for vibration spectra outlier detection. In *International Conference on Condition Monitoring and Machinery Failure Prevention Technologies*, pages 736–744, London, UK, 2012.
- [19] E. Hellinger. Neue Begründung der Theorie quadratischer Formen von unendlich vielen Veränderlichen. *Journal für die reine und angewandte Mathematik*, 136:210–271, 1909.
- [20] J. Ilonen, P. Paalanen, J. Kamarainen, and H. Kälviäinen. Gaussian mixture pdf in one-class classification: computing and utilizing confidence values. In *ICPR*, volume 2, pages 577–580, 2006.
- [21] KDD Cup. KDD Cup 1999 Data – Data Set. <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>, 1999. (last access: 06.02.2015).
- [22] S. Kullback and R. A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22:79–86, 1951.
- [23] M. Lichman. UCI machine learning repository, 2013.
- [24] M. Markou and S. Singh. Novelty Detection: a review – part 1: statistical approaches. *Signal Processing*, 83:2481–2497, 2003.
- [25] M. Markou and S. Singh. Novelty Detection: a review – part 2: neural network based approaches. *Signal Processing*, 83:2499–2521, 2003.
- [26] C. Müller-Schloer, H. Schmeck, and T. Ungerer. *Organic Computing – A Paradigm Shift for Complex Systems*. Springer-Verlag Berlin Heidelberg, 2011.
- [27] Papadimitriou, S. and Kitagawa, H. and Gibbons, P. B. and Faloutsos, C. LOCI: Fast Outlier Detection Using the Local Correlation Integral. In *Data Engineering*, volume 1, pages 315–326, 2003.
- [28] K. Pearson. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *Philosophical Magazine Series 5*, 50(302):157–175, 1900.
- [29] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko. A review of novelty detection. *Signal Processing*, 99:215–249, June 2014.
- [30] N. H. Pontoppidan and J. Larsen. Unsupervised condition change detection in large diesel engines. *Neural Networks for Signal Processing – Proc. of the IEEE XIII Workshop*, pages 565–574, 2003.
- [31] S. Roberts. Extreme value statistics for novelty detection in biomedical data processing. *IEE Proc. – Science, Measurement and Technology*, 147(6):363–367, 2000.
- [32] S. J. Roberts. Novelty detection using extreme value statistics. *Vision, Image and Signal Processing, IEEE Proc.*, 146(3):124–129, 1999.
- [33] E. J. Spinosa, F. de Carvalho, A. deLeon, and J. Gama. Novelty detection with application to data streams. *Intelligent Data Analysis*, 13(3):405–422, 2009.
- [34] L. Tarassenko, P. Hayton, N. Cerneaz, and M. Brady. Novelty detection for the identification of masses in mammograms. *Artificial Neural Networks, 1995., Fourth International Conference on*, (10):442–447, 1995.
- [35] D. Tax and R. Duin. Uniform Object Generation for Optimizing One-class Classifiers. *The Journal of Machine Learning Research*, 2:155–173, 2002.
- [36] C. H. Wang. Outlier identification and market segmentation using kernel-based clustering techniques. *Expert Systems with Applications*, 36(2):3744–3750, 2009.
- [37] D. Yeung and C. Chow. Parzen-window network intrusion detectors. In *Proc. of ICPR*, volume 4, pages 385–388, 2002.
- [38] F. Zorriassatine, A. Al-Habaibeh, R. M. Parkin, M. R. Jackson, and J. Coy. Novelty detection for practical pattern recognition in condition monitoring of multivariate processes: A case study. *International Journal of Advanced Manufacturing Technology*, 25(9-10):954–963, 2005.